Stateful Onion PIR

Offline

- Use Private Batched Subset Retrieval to get $\underbrace{S_1, \dots, S_c}_{\text{Downloaded to client local}}$, $|S_i| = k$ ⇐ optimized using Copy network (5.2)

why in FrodoPIR, $O(\sqrt{m}) = O(\sqrt{N})$ is required ?

Online

- Client find a set $S' \in \{S_1, \dots, S_c\}$ s.t. $i \notin S'$
- Client generate a partition $\begin{cases} P_1, \dots, P_m, \ m = \frac{N}{k+1}, \ |P| = k+1 \\ \text{and some } P_r \in \{P_1, \dots, P_m\} = S' \cup i, \ r \xleftarrow{\$} [m] \end{cases}$ $O(k)$
- Use stateless PIR to get P_r then calculate $\text{sum}(P_r) - \text{sum}(S')$ ⇒ value of i

FrodoPIR also stateful

Offline

- Server generate $A = PRG(u)$ and calculate $M = A \cdot DB$. Cheap
- Client download M, u, generate $\vec{b}, \vec{c}$

Download M is in $O(n \cdot \overset{\text{entry bit length}}{w}) = O(n \cdot \frac{w}{\log(\rho)})$

all configurable.

Online

- Client send query $\tilde{b} = \vec{b} + f_i$ to server → cheap
- Serval calculates $\tilde{c} = \tilde{b} \cdot DB$ and send back → matrix vector mult. Cheap.
- Client computes $\lfloor \tilde{c} - c \rceil$ → cheap